Attorney Docket No. 0023-0036

UNITED STATES PATENT APPLICATION

OF

Brian GAUDET

FOR

SYSTEMS AND METHODS FOR CONVERTING A *P* PACKET/CYCLE DATAPATH TO A *Q* PACKET/CYCLE DATAPATH

[0001] SYSTEMS AND METHODS FOR CONVERTING A *P* PACKET/CYCLE DATAPATH TO A *Q* PACKET/CYCLE DATAPATH

[0002] BACKGROUND OF THE INVENTION

[0003] Field of the Invention

[0004] The present invention relates generally to packet processing and, more particularly, to converting a first data path that carries up to P packets per processing cycle to a second data path that carries Q packets per processing cycle, where $Q < P$.

[0005] Description of Related Art

[0006] Packet processing systems, including any type of router, server or host that communicates using a packet-switching access mechanism, conventionally receive and process multiple packets in a single system cycle. A packet processing system may include a wide data path for receiving multiple packets in parallel during a single system cycle. Processing the packets at the rate they are received may require multiple instances of processing logic operating in parallel. If the packet processing system receive data path is N bytes wide, and the minimum packet that must be processed is M bytes, then $P = N/M$ instances of the processing logic may be required to process all packets in a given system cycle. For example, conventional Cyclical Redundancy Checking (CRC) may be performed to determine packet data errors.

[0007] Multiple instances of processing logic in the packet processing system, however, may have many drawbacks, such as increased power demands and space requirements in the system. In Application Specific Integrated Circuits (ASICs), for example, multiple instances of processing logic utilize valuable area of the ASIC. Multiple elements operating in parallel also increase timing complexity in the system.

[0008] To decrease space and power requirements in the packet processing system, it would, thus, be desirable to reduce the instances of the logic required to process multiple packets received during a single system cycle. For example, reduction of the processing logic to, for example, a single instance would significantly reduce space and power requirements. Therefore, to enable the use of a single instance of packet processing logic, there exists a need for systems and methods that can convert a data path carrying P packets per cycle to a data path that carries only Q packets per cycle, such as $Q = 1$ packet per cycle.

[0009] SUMMARY OF THE INVENTION

[0010] Consistent with the principles of the invention disclosed and claimed herein, these and other needs are addressed by queuing up to P packets per cycle received on a first data path and outputting Q packets per cycle on a second data path, where $Q < P$. Thus, in one embodiment, the packet processing system may use only a single instance of processing logic, such as, for example, CRC logic to process the P packets.

[0011] In accordance with the principles of the invention as embodied and broadly described herein, a method of converting a first data path carrying P packets per processing cycle to a second data path carrying N packets per processing cycle, wherein $N < P$, includes receiving the P packets during a first processing cycle on the first data path, storing the P packets in a queue; shifting first data from the queue into a shift register, selectively retrieving data from the shift register until a first set of Q packets of the P packets is retrieved, and sending the set of Q packets on the second data path during the first processing cycle.

[0012] Another implementation consistent with the principles of the invention may include determining whether the data in the shift register comprises an end-of-packet

indicator, a data field, and a start-of-packet indicator; and sending, based on the determination, a first set of Q packets on a second data path during the first processing cycle.

[0013] Yet another implementation consistent with the principles of the invention may include converting the *P* packets on the first data path to a first set of *Q* packets on the second data path, converting the *P* packets on the first data path to a second set of *Q* packets on the second data path, and processing the second set of *Q* packets on the second data path during a second processing cycle.

[0014] BRIEF DESCRIPTION OF THE DRAWINGS

[0015] The accompanying drawings, which are incorporated in and constitute a part of this specification, illustrate an embodiment of the invention and, together with the description, explain the invention. In the drawings,

[0016] FIG. 1 is an exemplary diagram of a packet processing system consistent with the principles of the invention;

[0017] FIG. 2 is an exemplary diagram of the First-in-First-Out (FIFO) queue and processing unit of FIG. 1 according to an implementation consistent with the principles of the invention; and

[0018] FIGS. 3-4 are exemplary flowcharts of processing by the packet processing system of FIG. 1 according to an implementation consistent with the principles of the invention.

[0019] DETAILED DESCRIPTION

[0020] The following detailed description of the invention refers to the accompanying drawings. The same reference numbers in different drawings may identify the same or

similar elements. Also, the following detailed description does not limit the invention. Instead, the scope of the invention is defined by the appended claims and equivalents.

[0021] Systems and methods consistent with the present invention provide mechanisms that can queue up to P packets per cycle received on a first data path and output Q packets per cycle on a second data path, wherein $Q < P$. In one embodiment, $Q = 1$, thus, enabling the use of a single instance of packet processing logic to process the received P packets.

[0022] EXEMPLARY PACKET PROCESSING SYSTEM

[0023] FIG. 1 is a diagram of an exemplary packet processing system 100 that converts a data path that may carry up to P packets per processing cycle to a data path that carries Q packets, such as a single packet, per processing cycle. Packet processing system 100 may reside in, or be connected to, any device that receives or transmits packet data, such as a packet router, a bridge, a switch fabric, or any server or host that handles packets.

[0024] Packet processing system 100 may include a data path 105, a FIFO queue 110, a processing unit 115, a data path 120, a processing device(s) 125, and a clock 130. Data path 105 may be N data units, such as N bytes, in width and may include conventional mechanisms for receiving up to P packets per processing cycle and for transmitting the received packets to FIFO queue 110. FIFO queue 110 can receive and store packets received from data path 105 and may include one or more memory devices (not shown).

[0025] Processing unit 115 may include a processor that contains instructions for processing data received from FIFO queue 110. Processing unit 115 may alternatively include logic circuitry that performs the equivalent functions of a processor. Processing unit 115 also may include flow control feedback to FIFO queue 110.

[0026] Processing device(s) 125 may include a device that processes up to Q incoming packets per processing cycle. For example, processing device(s) 125 may include a Cyclical Redundancy Check (CRC) unit that performs conventional CRC data error checks upon packets received via data path 105. Processing device(s) 125 may also include multiple devices that can each process a maximum of Q packets per cycle, where each of the multiple devices may perform different functions. For example, processing device(s) 125 may include a checksum unit, a CRC unit and a FIFO unit, with each unit processing a maximum of Q packets/cycle, such as 1 packet/cycle.

[0027] Clock 130 includes conventional circuitry for supplying clock signals to the components of packet processing system 100. Clock 130 may, for example, supply a clock signal to FIFO queue 110, processing unit 115, and processing device(s) 125.

[0028] EXEMPLARY FIFO QUEUE AND PROCESSING UNIT

[0029] FIG. 2 illustrates exemplary components of FIFO queue 110 and processing unit 115 according to an implementation consistent with the principles of the present invention. FIFO queue 110 may include memory locations 205 for storing bytes of packets received via data path 105. Processing unit 115 may include a shift register 210 and a control unit 215. Shift register 210 may include memory storage units that can right shift individual data units from FIFO queue 110 into shift register 210. Shift register 210 may include memory storage units for storing y data units. Control unit 215 may include a processing device that can retrieve data units from shift register 210 and pass individual packets, composed of the retrieved data units, on data path 120. Control unit 215 may alternatively include logic circuitry that performs the equivalent functions of a processing device.

[0030] EXEMPLARY PACKET PROCESSING

[0031] FIGS. 3-4 are exemplary flowcharts of processing by a system, such as packet processing system 100, according to an implementation consistent with the principles of the invention. Processing may begin by sending a clock (CLK) signal to start a processing cycle (step 305). Up to P packets may be received on a N byte data path during the processing cycle defined by the clock signal (step 310). The received packets may be stored in a FIFO queue (step 315). Byte counter c may be set to one (step 320) and it may be determined if byte c of the shift register indicates that it is a first byte of a packet (step 325). For example, byte c may include a start-of-packet (SOP) indicator. If byte c does not include an SOP indicator, the byte counter c may be incremented (step 330), and the process may return to step 325. If the byte c does include an SOP indicator, then byte c may be retrieved from the shift register (step 335).

[0032] Turning to FIG. 4, byte counter c may be incremented ($c = c + 1$) (step 405) and it may be determined if byte c in the shift register indicates that it is a last byte of a packet (step 410). For example, the byte may include an end-of-packet (EOP) indicator. If not, then it may be determined whether byte counter c is equal to the byte capacity y of the shift register (step 415). If byte counter c is equal to the maximum byte capacity (y) of the shift register, then byte c may be retrieved from the shift register (step 420) and y bytes may be right shifted from the FIFO queue into the shift register (step 425). Byte counter c may then be reset to one (step 430) and processing may return to step 335. If byte counter c is not equal to the byte storage capacity y of the shift register, then processing may return to step 335.

[0033] If byte c of the shift register indicates that it is the last byte of a packet, byte c may be retrieved from the shift register (step 435). A packet may then be sent including the

retrieved bytes on a 1 packet/cycle data path (step 440) at, or before, completion of the current processing cycle defined by the clock signal (step 445).

[0034] CONCLUSION

[0035] Consistent with the principles of the present invention, a data path carrying more than one packet per system cycle may be converted to a data path that carries only Q packets per system cycle, such as 1 packet/cycle. Thus, packet processing may be performed using only a single instance of processing logic, such as CRC logic, to process multiple packets received during a single system cycle.

[0036] The foregoing description of preferred embodiments of the present invention provides illustration and description, but is not intended to be exhaustive or to limit the invention to the precise form disclosed. Modifications and variations are possible in light of the above teachings or may be acquired from practice of the invention. For example, while series of steps have been described with regard to FIGS. 3-4, the order of the steps may differ or be performed in parallel in other implementations consistent with the present invention. Use of the "step" in the disclosure is not used in any functional sense, but rather refers to specific acts. Although apparatus, methods and other embodiments consistent with the principles of the invention may process incoming packets and bytes, other units of data may be processed without departing from the spirit and scope of the invention. For example, data entities other then packets can be processed and data units other than bytes can be used to transfer the data entities through the system.

[0037] The scope of the invention is defined by the claims and their equivalents.